사전 훈련된 딥 러닝 모델의 파인튜닝을 위한 최적의 배치 크기

아흐마드 이자즈, 신석주*
컴퓨터공학과, 조선대학교

ahmadijaz@chosun.kr, *sjshin@chosun.ac.kr (교신저자)

Optimal Batch Size for Fine-Tuning Pre-Trained Deep Learning Models

Ijaz Ahmad, Seokjoo Shin
Dept. of Computer Engineering, Chosun University.

요 약

Training a deep learning model from scratch significantly benefits from high resolution images. However, when using transfer learning, the higher resolution does not necessarily offer better performance and the model achieves better test accuracies on smaller images. The advantage of the smaller size is that it allows vast batch sizes during training, which can reduce model's training time. Therefore, in this study we have analyzed varying batch sizes when using a pre-trained MobileNetV2 model for classification task. We have performed our experiments on CIFAR 10 and CIFAR 100 datasets and the images are resized to 160x160. Our results have shown that the same or even improved accuracy can be achieved with less number of epochs when using larger batch size. For example, in the case of CIFAR10, there is no significant difference in the accuracy while in the case of CIFAR100 for batch size 512, there is a 2% improvement in the accuracy.

I. Introduction

Transfer learning (TL) is a machine learning method where what has been learned in one problem domain is repurposed to improve generalization in another related problem domain [1]. Usually, it is hard to get a dataset of sufficient size, e.g., medical image datasets, to train a network from scratch with random initialization. In such scenarios, TL improves the performance of a model through the transfer of knowledge and allows rapid progress. There are two main approaches of TL [2]. The first is to retrain only the classifier of the pre-trained model on the new dataset while its convolution network (ConvNet) is used as a fixed feature extractor. This approach is faster as only the classifier is trained. Alternatively, the second approach is to not only retrain the classifier but fine-tune the ConvNet on the new dataset as well. Fine-tuning a pre-trained CNN on the target data can significantly improve the performance of the model. However, when overfitting is of concern (e.g., when the new dataset is small), only a part of the ConvNet can be fine-tuned.

Transfer learning boosts the performance of a learning algorithm significantly; however, the pre-trained networks have certain constraints in terms of their architecture. For example, due to the wide availability of color images, the state-of-the-art pre-trained models are trained on color images. [3] has shown how to run pre-trained color image models on grayscale images. Their analysis has shown that comparable classification accuracy can be achieved when using grayscale images. The second challenge is that the available pre-trained models are trained for a specific input size. Therefore, require resizing the input images of different sizes.

The selection of optimal hyperparameter for training a deep learning model can avoid unnecessary computation [4]. The literature covers in detail the hyperparameter selection, e.g., image resolution, batch size, etc., when training neural networks from scratch. However, fewer studies have covered the selection of hyperparameter for transfer learning, e.g., [4] studied the momentum and [5] proposed Bayesian optimization to select learning rate for transfer learning. [6] studied the selection of optimal image resolution when fine-tuning a neural network. Their results have shown that the pre-trained models can achieve better accuracy on smaller size images. The smaller resolution of images can allow the use of larger batch sizes, which can reduce the training time. Therefore, in this work we have analyzed varying batch sizes when fine-tuning pre-trained MobileNetV2 model for classification task.

Ⅱ. Methods

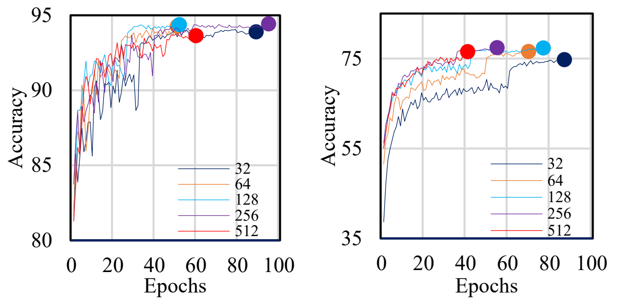
MobileNetV2 [7] is a deep learning model for classification, detection and segmentation tasks. The model is suitable to run on low computational devices such as mobile devices, thanks to its smaller number of parameters. The model uses separable convolution as efficient building blocks as its predecessors, but with two new features: linear bottlenecks between the layers and shortcut connections between the bottlenecks. The pre-trained MobileNetV2 comes with weights for five input image sizes (96, 128, 160, 192 and 224). When training MobileNetV2 model on ImageNet datasets from scratch, larger image sizes offer better performance. However, [6] has shown the contrast that when fine-tuning the pre-trained model for the classification of the CIFAR datasets, the model has better performance for image size of 160^2. Therefore, in our experiments we have used the pre-trained weights for input size of 160.

In our experiments, we have used the pre-trained MobileNetV2 model to classify CIFAR10 and CIFAR100 datasets [7]. In order to benefit from fine-tuning, we have initialized the ConvNet of the model with the pre-trained weights of MobileNetV2 and added a classifier on top of it as in [6]. The classifier has a global average-pooling layer followed by a drop out layer with ratio set to 0.5. The final layer is a dense layer and the number of nodes in the layer is equal to the number of classes in a dataset. We have perform our training in two stages: first, we have only retrain the classifier for 20 epochs. The initial learning rate was set to 10^{-2}. The learning was decremented by a factor of 10 when the validation accuracy stops improving. Fine-tuning a pre-trained CNN on the target data can significantly improve the performance of the model. Therefore, in the second stage of training, we have fine-tuned the whole network. The trained classifier from the previous stage and all the layers in the feature extractor were fine-tuned as well for 100 epochs. In the second stage, the initial learning rate was set to 10^{-4} be with a decrement factor of 10, in case the validation accuracy stops improving. In addition, to prevent the network from overfitting we have set early stopping criteria based on validation accuracy. During training, we have evaluated five batch sizes: 32, 64, 128, 256 and 512.

Ⅲ. Results and Discussion

In the simulations, we have used the MobileNetV2 pre-trained model as a classifier with the pre-trained weights for image resolution of 160x160. The experiments have been performed on the widely used CIFAR10 and CIFAR100 datasets [8]. The datasets consists of color images with resolution of 32x32. In our experiments, we have resized the images to 160x160 in order to be same as the network input size. Both datasets have 60k images with 50k training images and 10k test images. CIFAR10 has 10 classes and 6k images per class, while CIFAR100 has 100 classes and 600 images per class. In experiments, we have used 10% of training images as validation images.

To demonstrate the relationship between the batch size and classification accuracy, we have considered five batch sizes (32, 64, 128, 256 and 512) as shown in Fig. 1. We have plotted the validation accuracy as a function of the number of epochs for CIFAR10 in Fig. 1. (a) and CIFAR100 in Fig. 1. (b). It can be seen that varying batch sizes requires different number of epochs shown by the circle on the end of the line. The larger batch size requires fewer numbers of epochs than the smaller batch sizes. The abrupt change in the accuracy is the result of the learning rate decay. For CIFAR10, there is no significant difference in the accuracy for varying batch sizes. However, in the case of CIFAR100 for batch size 512, there is a 2% improvement in the accuracy. In addition, the benefits of the transfer learning can also be observed during training as the



(a) (b)

Fig. 1. Validation accuracy of pre-trained MobileNetV2 on (a) CIFAR10 and (b) CIFAR100 for different batch sizes.

TABLE I. Classification accuracy of the pre-trained MobileNetV2 model on different batch sizes.

Batch Sizes	**CIFAR10**	**CIFAR100**
32	93.83	74.32
64	93.97	76.26
128	93.88	76.71
256	93.66	76.97
512	93.53	76.49

model starts with a better accuracy. We have summarized the achieved accuracy of MobileNetV2 on the CIFAR datasets for different batch sizes in Table I.

IV. Conclusion

The selection of optimal hyperparameter for training a deep learning model can avoid unnecessary computation. The literature extensively convers hyperparameter selection when training neural networks from scratch. However, there are few studies that cover the hyperparameter for transfer learning. Therefore, in this study we have analyzed various batch sizes for fine-tuning a pre-trained model. Our results have shown that the same accuracy can be achieved in fewer numbers of epochs with larger batch sizes.

Acknowledgment

This research is supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF-2018R1D1A1B07048338).

References

[1] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. Cambridge, Massachusetts: The MIT Pres, 2016.

[2] "CS231n Convolutional Neural Networks for Visual Recognition." https://cs231n.github.io/transfer-learning/ (Dec. 18, 2020).

[3] I. Ahmad and S. Shin, "An Approach to Run Pre-Trained Deep Learning Models on Grayscale Images," in *2021 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, Jeju, Korea, Apr. 2021, pp. 1-4.

[4] H. Li *et al.*, "Rethinking the Hyperparameters for Fine-tuning," *ArXiv*, vol. abs/2002.11770, 2020.

[5] R. J. Borgli, H. Kvale Stensland, M. A. Riegler, and P. Halvorsen, "Automatic Hyperparameter Optimization for Transfer Learning on Medical Image Datasets Using Bayesian Optimization," in *2019 13th ISMICT*, Oslo, Norway, May 2019, pp. 1-6. doi: 10.1109/ISMICT.2019.8743779.

[6] I. Ahmad and S. Shin, "Optimal resolution selection to run pre-trained deep learning models on tiny images," in *2021 한국차세대컴퓨팅학회 춘계학술대회*, Korea, May 21, pp. 172-174.

[7] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks,"

ArXiv180104381 Cs, Mar. 2019, Accessed: Mar. 08, 2021. [Online]. Available: http://arxiv.org/abs/1801.04381
[8] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," p. 60.